# Counting Environments and Closures

**Maciej Bendkowski**[1]
Jagiellonian University,
Faculty of Mathematics and Computer Science,
Theoretical Computer Science Department,
ul. Prof. Łojasiewicza 6, 30–348 Kraków, Poland
maciej.bendkowski@tcs.uj.edu.pl

**Pierre Lescanne**
University of Lyon, École normale supérieure de Lyon,
LIP (UMR 5668 CNRS ENS Lyon UCBL),
46 allée d'Italie, 69364 Lyon, France
pierre.lescanne@ens-lyon.fr

## Abstract

Environments and closures are two of the main ingredients of evaluation in lambda-calculus. A closure is a pair consisting of a lambda-term and an environment, whereas an environment is a list of lambda-terms assigned to free variables. In this paper we investigate some dynamic aspects of evaluation in lambda-calculus considering the quantitative, combinatorial properties of environments and closures. Focusing on two classes of environments and closures, namely the so-called plain and closed ones, we consider the problem of their asymptotic counting and effective random generation. We provide an asymptotic approximation of the number of both plain environments and closures of size $n$. Using the associated generating functions, we construct effective samplers for both classes of combinatorial structures. Finally, we discuss the related problem of asymptotic counting and random generation of closed environments and closures.

**2012 ACM Subject Classification** Mathematics of computing → Lambda calculus, Mathematics of computing → Generating functions

**Keywords and phrases** lambda-calculus, combinatorics, functional programming, mathematical analysis, complexity

**Digital Object Identifier** 10.4230/LIPIcs.FSCD.2018.11

## 1 Introduction

Though, traditionally, computational complexity is investigated in the context of Turing machines since their initial development, evaluation complexity in various term rewriting systems, such as $\lambda$-calculus or combinatory logic, attracts increasing attention only quite recently. For instance, let us mention the worst-case analysis of evaluation, based on the invariance of unitary cost models [26, 3, 1] or transformation techniques proving termination of term rewriting systems [2].

Much like in classic computational complexity, the corresponding average-case analysis of evaluation in term rewriting systems follows a different, more combinatorial and quantitative approach, compared to its worst-case variant. In [10, 11] Choppy, Kaplan and Soria propose an average-case complexity analysis of normalisation in a general class of term rewriting systems using generating functions, in particular techniques from analytic combinatorics [19].

[1] Maciej Bendkowski was partially supported within the Polish National Science Center grant 2016/21/N/ST6/01032.


Following a somewhat similar path, Bendkwoski, Grygiel and Zaionc investigated later the asymptotic properties of normal-order reduction in combinatory logic, in particular the normalisation cost of large random combinators [7, 4]. Alas, normalisation in $\lambda$-calculus has not yet been studied in such a combinatorial context. Nonetheless, static, quantitative properties of $\lambda$-terms, form an active stream of recent research. Let us mention, non-exhaustively, investigations into the asymptotic properties of large random $\lambda$-terms [15, 6] or their effective counting and random generation ensuring a uniform distribution among terms with equal size [8, 23, 22, 9].

In the current paper, we take a step towards the average-case analysis of reduction complexity in $\lambda$-calculus. Specifically, we offer a quantitative analysis of environments and closures — two types of structures frequently present at the core of abstract machines modelling $\lambda$-term evaluation, such as for instance the Krivine or U- machine [13, 28]. In Section 3 we discuss the combinatorial representation of environments and closures, in particular the associated de Bruijn notation. In Section 4 we list the analytic combinatorics tools required for our analysis. Next, in Section 5 and Section 6 we conduct our quantitative investigation into so-called plain and closed environments and closures, respectively, subsequently concluding the paper in Section 7.

## 2 A combinatoric approach to higher order rewriting systems

As said in the introduction, viewing the $\lambda$-calculus from the perspective of counting is new, especially in the scientific community of structures for computation and deduction and requires motivation to be detailed.

First, clearly a new perspective on $\lambda$-calculus enlightens the semantics and opens new directions, especially by adding a touch of efficiency and a discussion on how the size of structures with binders (like $\lambda$-terms) can be measured. However, despite advanced mathematical techniques are used, the goal is more practical and connected to operational semantics and implementation. Counting allows assigning a precise measure on how a specific algorithm performs. In [24][2] Knuth calls analysis of Type A an *analysis of a particular algorithm* and shows how important it is in computer science. He adds (p. 3): "Complexity analysis provides an interesting way to sharpen our tools for the more routine problems we face from day to day."

Furthermore, a notion of probabilistic distribution as used in the average-case analysis of algorithm, after Sedgewick and Flajolet [35], is deduced. In particular a notion of uniform distribution is inferred in order to evaluate the *average case efficiency* of algorithms w.r.t. this distribution. In this paper, the algorithms the authors have in mind are the several reduction machines for the $\lambda$-calculus, especially the Krivine machine and the U-machine, for which analyses of Type A and more specifically average case analyses are expected to be built. Another application is *random generation* of terms and several kinds of structures for computation and deduction as used for instance in QuickCheck [12]. A fully and mathematically justified random generator can only be built using the kind of tools developed in this paper.

But average case analysis based on uniform distribution is not the only one. The so-called *smoothed analysis of algorithms* [36] is another family of tools which is based on measures of size. Here the distribution is no more uniform and this method has promising applications, hopefully in structures for computation and deduction.

[2] This paper is part of the book "Selected Papers on Analysis of Algorithms" [25] dedicated to Professor N. G. de Bruijn.

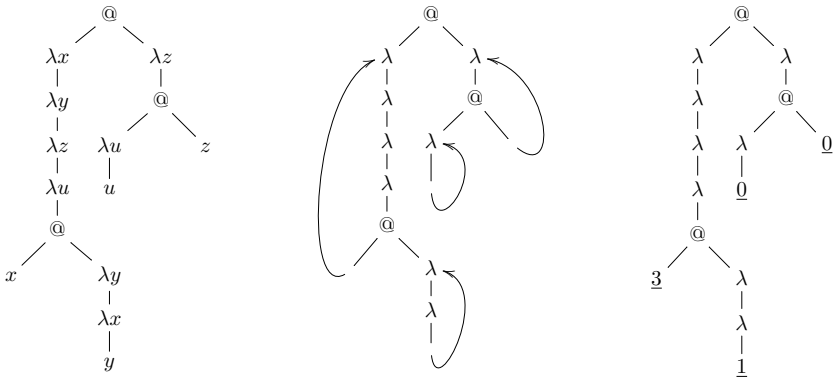


**Figure 1** Three representations of the $\lambda$-term $T = (\lambda xyzu.x(\lambda yx.y))\ (\lambda z.(\lambda u.u)z)$.

## 3 Environments and closures

In this section we outline the de Bruijn notation and related concepts deriving from $\lambda$-calculus variants with explicit substitutions used in the subsequent sections.

### 3.1 De Bruijn notation

Though the classic variable notation for $\lambda$-terms is elegant and concise, it poses considerable implementation issues, especially in the context of substitution resolution and potential name clashes. In order to accommodate these problems, de Bruijn proposed an alternative name-free notation for $\lambda$-terms [16]. In this notation, each variable $x$ is replaced by an appropriate non-negative integer $\underline{n}$ (so-called *index*) intended to encode the distance between $x$ and its binding abstraction. Specifically, if $x$ is bound to the $(n+1)$st abstraction on its unique path to the term root in the associated $\lambda$-tree, then $x$ is replaced by the index $\underline{n}$. In this manner, each closed $\lambda$-term in the classic variable notation is representable in the de Bruijn notation.

▶ **Example 1.** Consider the $\lambda$-term $T = (\lambda xyzu.x(\lambda yx.y))\ (\lambda z.(\lambda u.u)z)$. Figure 1 depicts three different representations of $T$ as tree-like structures. The first one uses explicit variables, the second one uses back pointers to represent the bound variables, whereas the third one uses de Bruijn indices.

In order to represent free occurrences of variables, one uses indices of values exceeding the number of abstractions crossed on respective paths to the term root. For instance, $\lambda x.yz$ can be represented as $\lambda\underline{12}$ since $\underline{1}$ and $\underline{2}$ correspond to two different variable occurrences.

Recall that in the classic variable notation a $\lambda$-term $M$ is said to be *closed* if each of its variables is bound. In the de Bruijn notation, it means that for each index occurrence $\underline{n}$ in $M$ one finds at least $n+1$ abstractions on the unique path from $\underline{n}$ to the term root of $M$. If a $\lambda$-term is not closed, it is said to be *open*. If heading $M$ with $m$ abstractions turns it into a closed $\lambda$-term, then $M$ is said to be *$m$-open*. In particular, closed $\lambda$-terms are 0-open.

▶ **Example 2.** Note that $\lambda\lambda\lambda\lambda(\underline{3}(\lambda\lambda\underline{1}))\ (\lambda(\lambda\underline{0})\underline{0})$ is closed. The $\lambda$-term $\underline{3}(\lambda\lambda\underline{1})$ is 4-open, however it is not 3-open. Indeed, $\lambda\lambda\lambda(\underline{3}(\lambda\lambda\underline{1}))$ is 1-open instead of being closed. Similarly, $\lambda(\underline{3}(\lambda\lambda\underline{1}))$ is 3-open, however it is not 2-open.

Certainly, the set $\mathcal{L}_m$ of $m$-open terms is a subset of the set of $(m+1)$-open terms. In other words, if $M$ is $m$-open, it is also $(m+1)$-open. The set of all $\lambda$-terms is called the set of *plain* terms. It is the union of the sets of $m$-open terms and is denoted as $\mathcal{L}_\infty$. Hence,

$$\mathcal{L}_0 \subseteq \mathcal{L}_1 \subseteq \cdots \subseteq \mathcal{L}_m \subseteq \mathcal{L}_{m+1} \cdots \subseteq \bigcup_{i=0}^{\infty} \mathcal{L}_i = \mathcal{L}_\infty \,. \tag{1}$$

Let us note that de Bruijn's name-free representation of $\lambda$-terms exhibits an important combinatorial benefit. Specifically, each $\lambda$-term in the de Bruijn notation represents an entire $\alpha$-equivalence class of $\lambda$-terms in the classical variable notation. Indeed, two variable occurrences bound by the same abstraction are assigned the same de Bruijn index. In consequence, counting $\lambda$-terms in the de Bruijn notation we are, in fact, counting entire $\alpha$-equivalence classes instead of their inhabitants.

## 3.2 Closures and $\beta$-reduction

Recall that the main rewriting rule of $\lambda$-calculus is *$\beta$-reduction*, see, e.g. [14]

$$(\beta) \quad (\lambda M)\ N \quad \to \quad M\{\underline{0} \leftarrow N\} \tag{2}$$

where the operation $\{\underline{n} \leftarrow M\}$, i.e. substitution of $\lambda$-terms for de Bruijn indices, is defined inductively as follows:

$$\begin{aligned}
(M\ N)\{\underline{n} \leftarrow P\} &= M\{\underline{n} \leftarrow P\}\ N\{\underline{n} \leftarrow P\} \\
(\lambda M)\{\underline{n} \leftarrow P\} &= \lambda(M\{\underline{(n+1)} \leftarrow P\}) \\
\underline{m}\{\underline{n} \leftarrow P\} &= \begin{cases} \underline{m-1} & \text{if } m > n \\ \tau_0^n(P) & \text{if } m = n \\ \underline{m} & \text{if } m < n\,. \end{cases}
\end{aligned} \tag{3}$$

Th first rule distributes the substitution in an application, the second rule pushes a substitution under an abstraction and the third rule tells how a substitution acts when the term is an index. $\tau_0^n(P)$ tells how to update the indices of a term which is substituted for an index. The operation $\tau_i^n(M)$ is defined by induction on $M$ as

$$\begin{aligned}
\tau_i^n(M\ N) &= \tau_i^n(M)\ \tau_i^n(N) \\
\tau_i^n(\lambda M) &= \lambda(\tau_{i+1}^n(M)) \\
\tau_i^n(\underline{m}) &= \begin{cases} \underline{m+n-1} & \text{if } m > i \\ \underline{m} & \text{if } m \le i\,. \end{cases}
\end{aligned} \tag{4}$$

A $\lambda$-term in the form of $(\lambda M)\ N$ is called a *$\beta$-redex* (or simply a *redex*). Lambda terms not containing $\beta$-redexes as subterms, are called ($\beta$-)normal forms. The computational process of rewriting (reducing) a $\lambda$-term to its $\beta$-normal form by successive elimination of $\beta$-redexes is called *normalisation*. There exists an abundant literature on normalisation in $\lambda$-calculus; let us mention, not exhaustively [27, 33, 29, 13, 30].

One of the central concepts present in various formalisms dealing with normalisation in $\lambda$-calculus are environments and closures. An *environment* is a list of values meant to be assigned to indices $\underline{0}, \underline{1}, \underline{2}, \ldots, \underline{m-1}$ of an $m$-open $\lambda$-term. A *closure*, on the other hand, is a couple consisting of a $\lambda$-term and an environment. Such couples are meant to represent closed, not yet fully evaluated, $\lambda$-terms. For instance, the closure $\langle M, \square \rangle$ consists of the

$\lambda$-term $M$ evaluated in the context of an empty environment, denoted as $\Box$, and represents simply $M$. The closure $\langle \underline{1}\,\underline{0}, \langle \lambda\underline{0}, \Box\rangle : \langle \lambda\lambda\underline{0}, \Box\rangle : \Box\rangle$ represents the $\lambda$-term $(\underline{1}\,\underline{0})$ evaluated in the context of an environment $\langle \lambda\underline{0}, \Box\rangle : \langle \lambda\lambda\underline{0}, \Box\rangle : \Box$. Here, intuitively, the index $\underline{1}$ is receiving the value $\lambda\underline{0}$ whereas the index $\underline{0}$ is being assigned $\lambda\lambda\underline{0}$. Finally, $\lambda\underline{0}$ is applied to $\lambda\lambda\underline{0}$. And so, reducing the closure $\langle \underline{1}\,\underline{0}, \langle \lambda\underline{0}, \Box\rangle : \langle \lambda\lambda\underline{0}, \Box\rangle : \Box\rangle : \Box$, for instance using a Krivine abstract machine [13], we obtain $\lambda\lambda\underline{0}$.

Let us notice that following the outlined description of environments and closures, we can provide a formal combinatorial specification for both using the following mutually recursive definitions:

$$\begin{aligned} \mathcal{C}los &::= \langle \Lambda, \mathcal{E}nv\rangle \\ \mathcal{E}nv &::= \Box \mid \mathcal{C}los : \mathcal{E}nv \end{aligned} \tag{5}$$

In the above specification, $\Lambda$ denotes the set of all plain $\lambda$-terms. Moreover, we introduce two binary operators "$\langle \_, \_\rangle$", i.e. the *coupling* operator, and ":", i.e. the *cons* operator, heading its left-hand side on the right-hand list. When applied to a $\lambda$-term and an environment, the coupling operator constructs a new closure. In other words, a *closure* is a couple of a $\lambda$-term and an environment whereas an environment is a list of closures, representing a list of assignments to free occurrences of de Bruijn indices.

Such a combinatorial specification for closures and environments plays an important rôle as it allows us to investigate, using methods of analytic combinatorics, the quantitative properties of both closures and environments.

## 4 Analytic tools

In the following section we briefly[3] outline the main techniques and notions from the theory of generating functions and singularity analysis. We refer the curious reader to [19, 37, 21] for a thorough introduction.

Let $(f_n)_n$ be a sequence of non-negative integers. Then, the *generating function* $F(z)$ associated with $(f_n)_n$ is the formal power series $F(z) = \sum_{n\geq 0} f_n z^n$. Following standard notational conventions, we use $[z^n]F(z)$ to denote the coefficient standing by $z^n$ in the power series expansion of $F(z)$. Given two sequences $(a_n)_n$ and $(b_n)_n$ we write $a_n \sim b_n$ to denote the fact that both sequences admit the same asymptotic growth order, specifically $\lim_{n\to\infty} \frac{a_n}{b_n} = 1$. Finally, we write $\varphi \doteq c$ if we are interested in the numerical approximation $c$ of an expression $\varphi$.

Suppose that $F(z)$, viewed as a function of a single complex variable $z$, is defined in some region $\Omega$ of the complex plane centred at $z_0 \in \Omega$. Then, if $F(z)$ admits a convergent power series expansion in form of

$$F(z) = \sum_{n\geq 0} f_n (z - z_0)^n \tag{6}$$

it is said to be *analytic* at point $z_0$. Moreover, if $F(z)$ is analytic at each point $z \in \Omega$, then $F(z)$ is said to be *analytic in the region* $\Omega$. Suppose that there exists a function $G(z)$ analytic in a region $\Omega^*$ such that $\Omega \cap \Omega^* \neq \emptyset$ and both $F(z)$ and $G(z)$ agree on $\Omega \cap \Omega^*$, i.e. $F|_{\Omega\cap\Omega^*} = G|_{\Omega\cap\Omega^*}$. Then, $G(z)$ is said to be an *analytic continuation* of $F(z)$ onto $\Omega^*$. If

[3] In such a short presentation of a non-trivial theory, many terms, like "branch", "Newton-Puiseux series", "locally convergent" etc. are not defined. They are defined in the references [19, 37, 21].

$F(z)$ defined in some region $\Omega \setminus \{z_0\}$ has no analytic continuation onto $\Omega$, then $z_0$ is said to be a *singularity* of $F(z)$. When a formal power series $F(z) = \sum_{n\geq 0} f_n z^n$ represents an analytic function in some neighbourhood of the complex plane origin, it becomes possible to link the location and type of singularities corresponding to $F(z)$, in particular so-called *dominating* singularities residing at the respective circle of convergence, with the asymptotic growth rate of its coefficients. This process of *singularity analysis* developed by Flajolet and Odlyzko [18] provides a general and systematic technique for establishing the quantitative aspects of a broad class of combinatorial structures.

While investigating environments and closures, a particular example of algebraic combinatorial structures, the respective generating functions turn out to be algebraic themselves. The following prominent tools provide the essential foundation underlying the process of *algebraic singularity analysis* based on *Newton-Puiseux expansions*, i.e. extensions of power series allowing fractional exponents.

▶ **Theorem 3** (Newton, Puiseux [19, Theorem VII.7]). *Let $F(z)$ be a branch of an algebraic equation $P(z, F(z)) = 0$. Then, in a circular neighbourhood of a singularity $\rho$ slit along a ray emanating from $\rho$, $F(z)$ admits a fractional Newton-Puiseux series expansion that is locally convergent and of the form*

$$F(z) = \sum_{k\geq k_0} c_k (z-\rho)^{k/\kappa} \tag{7}$$

*where $k_0 \in \mathbb{Z}$ and $\kappa \geq 1$.*

Let $F(z)$ be analytic at the origin. Note that $[z^n]F(z) = \rho^{-n}[z^n]F(\rho z)$. In consequence, following a proper rescaling we can focus on the type of singularities of $F(z)$ on the unit circle. The standard function scale provides then the asymptotic expansion of $[z^n]F(z)$.

▶ **Theorem 4** (Standard function scale [19, Theorem VI.1]). *Let $\alpha \in \mathbb{C} \setminus \mathbb{Z}_{\leq 0}$. Then, $F(z) = (1-z)^{-\alpha}$ admits for large $n$ a complete asymptotic expansion in form of*

$$[z^n]F(z) = \frac{n^{\alpha-1}}{\Gamma(\alpha)}\left(1 + \frac{\alpha(\alpha-1)}{2n} + \frac{\alpha(\alpha-1)(\alpha-2)(3\alpha-1)}{24n^2} + O\left(\frac{1}{n^3}\right)\right) \tag{8}$$

*where $\Gamma\colon \mathbb{C} \setminus \mathbb{Z}_{\leq 0} \to \mathbb{C}$ is the Euler Gamma function defined as*

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx \qquad \text{for } \Re(z) > 0 \tag{9}$$

*and by analytic continuation on all its domain.*

Given an analytic generating function $F(z)$ implicitly defined as a branch of an algebraic function $P(z, F(z)) = 0$ our task of establishing the asymptotic expansion of the corresponding sequence $([z^n]F(z))_n$ reduces therefore to locating and studying the (dominating) singularities of $F(z)$. For generating functions analytic at the complex plane origin, this quest simplifies even further due to the following classic result.

▶ **Theorem 5** (Pringsheim [19, Theorem IV.6]). *If $F(z)$ is representable at the origin by a series expansion that has non-negative coefficients and radius of convergence $R$, then the point $z = R$ is a singularity of $F(z)$.*

We can therefore focus on the real line while searching for respective singularities. Since $\sqrt{z}$ cannot be unambiguously defined as an analytic function at $z = 0$ we primarily focus on roots of radicand expressions in the closed-form formulae of investigated generating functions.

## 4.1 Counting λ-terms

Let us outline the main quantitative results concerning λ-terms in the de Bruijn notation, see [6, 22]. In this combinatorial model, indices are represented in a unary encoding using the successor operator $\mathsf{S}$ and 0. In the so-called *natural* size notion [6], assumed throughout the current paper, the size of λ-terms is defined recursively as follows:

$$\begin{array}{lcllcl} |0| & = & 1 & |M\,N| & = & |M|+|N|+1 \\ |\mathsf{S}\ n| & = & |\underline{n}| \ = \ |n|+1 & |\lambda M| & = & |M|+1\,. \end{array}$$

And so, for example, $|\lambda\underline{12}|= 7$. We briefly remark that different size notions in the de Bruijn representation, alternative to the assumed natural one, are considered in the literature. Though all share similar asymptotic properties, we choose to consider the above size notion in order to minimise the technical overhead of the overall presentation. We refer the curious reader to [22, 9] for a detailed analysis of various size notions in the de Bruijn representation.

Let $l_n$ denote the number of plain λ-terms of size $n$. Consider the generating function $L_\infty(z) = \sum_{n\geq 0} l_n z^n$. Using symbolic methods, see [19, Part A. Symbolic Methods] we note that $L_\infty(z)$ satisfies

$$L_\infty(z) \ = \ zL_\infty(z) + zL_\infty(z)^2 + D(z) \qquad \text{where} \qquad D(z) = \frac{z}{1-z} = \sum_{n=0}^{\infty} z^{n+1}. \tag{10}$$

In words, a λ-term is either (a) an abstraction followed by another λ-term, accounting for the first summand, (b) an application of two λ-terms, accounting for the second summand, or finally, (c) a de Bruijn index which is, in turn, a sequence of successors applied to 0. Solving (10) for $L_\infty(z)$ we find that the generating function $L_\infty(z)$, taking into account that the coefficients $l_n$ are positive for all $n$, admits the following closed-form solution:

$$L_\infty(z) = \frac{1-z-\sqrt{(1-z)^2-\frac{4z}{1-z}}}{2z}\,. \tag{11}$$

In such a form, $L_\infty(z)$ is amenable to the standard techniques of singularity analysis. In consequence we have the following general asymptotic approximation of $l_n$.

▶ **Theorem 6** (Bendkowski, Grygiel, Lescanne, Zaionc [6]). *The sequence $([z^n]L_\infty(z))_n$ corresponding to plain λ-terms of size $n$ admits the following asymptotic approximation:*

$$[z^n]L_\infty(z) \sim C\rho_{L_\infty}^{-n} n^{-3/2} \tag{12}$$

*where*

$$\rho_{L_\infty} = \frac{1}{3}\left(\sqrt[3]{26+6\sqrt{33}} - \frac{4\ 2^{2/3}}{\sqrt[3]{13+3\sqrt{33}}} - 1\right) \doteq 0.29559 \quad \textit{and} \quad C \doteq 0.60676. \tag{13}$$

In the context of evaluation, the arguably most interesting subclass of λ-terms are closed or, more generally, $m$-open λ-terms. Recall that an $m$-open λ-term takes one of the following forms. Either it is (a) an abstraction followed by an $(m+1)$-open λ-term, or (b) an application of two $m$-open λ-terms, or finally, (c) one of the indices $\underline{0}, \underline{1}, \ldots, \underline{m-1}$. Such a specification for $m$-open λ-terms yields the following functional equation defining the associated generating function $L_m(z)$:

$$L_m(z) = zL_{m+1}(z) + zL_m(z)^2 + \frac{1-z^m}{1-z}\,. \tag{14}$$

Since $L_m(z)$ depends on $L_{m+1}(z)$, solving (14) for $L_m(z)$ one finds that

$$L_m(z) = \frac{1 - \sqrt{1 - 4z^2\left(L_{m+1}(z) + \frac{1-z^m}{1-z}\right)}}{2z}\,. \tag{15}$$

Such a presentation of $L_m(z)$ poses considerable difficulties as $L_m(z)$ depends on $L_{m+1}(z)$ depending itself on $L_{m+2}(z)$, etc. If developed, the formula (15) for $L_m(z)$ consists of an infinite number of nested radicals. In consequence, standard analytic combinatorics tools do not provide the asymptotic expansion of $[z^n]L_m(z)$, in particular $[z^n]L_0(z)$ associated with closed $\lambda$-terms. In their recent breakthrough paper, Bodini, Gittenberger and Gołębiewski [9] propose a clever approximation of the infinite system associated with $L_m(z)$ and give the following asymptotic approximation for the number of $m$-open $\lambda$-terms.

▶ **Theorem 7** (Bodini, Gittenberger and Gołębiewski [9]). *The sequence $([z^n]L_m(z))_n$ corresponding to $m$-open $\lambda$-terms of size $n$ admits the following asymptotic approximation:*

$$[z^n]L_m(z) \sim C_m \rho_{L_\infty}^{-n} n^{-3/2} \tag{16}$$

*where $\rho_{L_\infty}$ is the dominant singularity corresponding to plain $\lambda$-terms, see* (13)*, and $C_m$ is a constant, depending solely on $m$.*

Let us remark that for closed $\lambda$-terms, the constant $C_0$ lies in between 0.07790995266 and 0.0779099823. In what follows, we use the above Theorem 7 in our investigations regarding what we call closed closures.

## 5 Counting plain closures and environments

In this section we start with counting *plain environments and closures*, i.e. members of $\mathcal{E}nv$ and $\mathcal{C}los$, see (5). We consider a simple model in which the size of environments and closures is equal to the total number of abstractions, applications and the sum of all the de Bruijn index sizes. Formally, we set

$$|\langle M, e\rangle| = |M| + |e| \qquad |\mathfrak{c} : e| = |\mathfrak{c}| + |e| \qquad |\Box| = 0\,.$$

▶ **Example 8.** The following two tables list the first few plain environments and closures.

| size | environments | total |
|---|---|---|
| 0 | $\Box$ | 1 |
| 1 | $\langle \underline{0}, \Box\rangle : \Box$ | 1 |
| 2 | $\langle \underline{0}, \Box\rangle : \langle \underline{0}, \Box\rangle : \Box$<br>$\langle \underline{0}, \langle \underline{0}, \Box\rangle : \Box\rangle : \Box$<br>$\langle \lambda\underline{0}, \Box\rangle : \Box, \quad \langle \underline{1}, \Box\rangle : \Box$ | 4 |

| size | closures | total |
|---|---|---|
| 0 | | 0 |
| 1 | $\langle \underline{0}, \Box\rangle$ | 1 |
| 2 | $\langle \underline{0}, \langle \underline{0}, \Box\rangle\rangle$<br>$\langle \lambda\underline{0}, \Box\rangle \quad \langle \underline{1}, \Box\rangle$ | 3 |

By analogy with the notation $\mathcal{L}_\infty$ for the set of plain $\lambda$-terms, we write $\mathcal{E}_\infty$ and $\mathcal{C}_\infty$ to denote the class of plain environments and closures, respectively. Reformulating (5) we can now give a formal specification for both $\mathcal{E}_\infty$ and $\mathcal{C}_\infty$ as follows:

$$\begin{aligned} \mathcal{E}_\infty &= \mathcal{C}_\infty : \mathcal{E}_\infty \;\mid\; \Box \\ \mathcal{C}_\infty &= \langle \mathcal{L}_\infty, \mathcal{E}_\infty\rangle\,. \end{aligned} \tag{17}$$

In such a form, both classes $\mathcal{E}_\infty$ and $\mathcal{C}_\infty$ become amenable to the process of singularity analysis. In consequence, we obtain the following asymptotic approximation for the number of plain environments and closures.

▶ **Theorem 9.** *The numbers $e_n$ and $c_n$ of plain environments and closures of size $n$, respectively, admit the following asymptotic approximations:*

$$e_n \sim C_e \cdot \rho^{-n} n^{-3/2} \quad \text{and} \quad c_n \sim C_c \cdot \rho^{-n} n^{-3/2} \tag{18}$$

*where*

$$C_e = \frac{\sqrt{\frac{5}{47}\left(109 + 35\sqrt{545}\right)}}{8\sqrt{\pi}} \doteq 0.699997,$$
$$C_c = \frac{\sqrt{\frac{10\left(48069\sqrt{5} - 10295\sqrt{109}\right)}{65\sqrt{109} - 301\sqrt{5}}}}{\sqrt{\pi}\left(77 - 3\sqrt{545}\right)} \doteq 0.174999 \tag{19}$$

*and*

$$\rho = \frac{1}{10}\left(25 - \sqrt{545}\right) \doteq 0.165476 \quad \text{giving} \quad \rho^{-n} \doteq 6.04315^n. \tag{20}$$

**Proof.** Consider generating functions $E_\infty(z)$ and $C_\infty(z)$ associated with respective counting sequences, i.e. the sequence $(e_n)_n$ of plain environments of size $n$ and $(c_n)_n$ of plain closures of size $n$. Based on the specification (17) for $\mathcal{E}_\infty$ and $\mathcal{C}_\infty$ and the assumed size notion, we can write down the following system of functional equations satisfied by $E_\infty(z)$ and $C_\infty(z)$:

$$\begin{aligned} E_\infty(z) &= C_\infty(z) E_\infty(z) + 1 \\ C_\infty(z) &= L_\infty(z)\, E_\infty(z). \end{aligned} \tag{21}$$

Next, we solve (21) for $E_\infty(z)$ and $C_\infty(z)$. Though (21) has two formal solutions, the following one is the single one yielding analytic generating functions with non-negative coefficients:

$$E_\infty(z) = \frac{1 - \sqrt{1 - 4L_\infty(z)}}{2L_\infty(z)} \quad \text{and} \quad C_\infty(z) = \frac{1}{2}\left(1 - \sqrt{1 - 4L_\infty(z)}\right). \tag{22}$$

Since $L_\infty(z) > 0$ for $z \in (0, \rho_{L_\infty})$ there are two potential sources of singularities in (22). Specifically, the dominating singularity $\rho_{L_\infty}$ of $L_\infty(z)$, see (13), or roots of the radicand expression $1 - 4L_\infty(z)$. Therefore, we have to determine whether we fall into the so-called sub- or super-critical composition schema, see [19, Chapter VI. 9]. Solving $1 - 4L_\infty(z) = 0$ for $z$, we find that it admits a single solution $\rho$ equal to

$$\rho = \frac{1}{10}\left(25 - \sqrt{545}\right) \doteq 0.165476\,. \tag{23}$$

Since $\rho < \rho_{L_\infty}$ the outer radicand carries the dominant singularity $\rho$ of both $E_\infty(z)$ and $C_\infty(z)$. We fall therefore directly into the super-critical composition schema and in consequence know that near $\rho$ both $E_\infty(z)$ and $C_\infty(z)$ admit Newton-Puiseux expansions in form of

$$E_\infty(z) = a_{E_\infty} + b_{E_\infty}\sqrt{1 - \frac{z}{\rho}} + O\left(\left|1 - \frac{z}{\rho}\right|\right)$$

and

$$C_\infty(z) = a_{C_\infty} + b_{C_\infty}\sqrt{1 - \frac{z}{\rho}} + O\left(\left|1 - \frac{z}{\rho}\right|\right) \tag{24}$$

with $a_{E_\infty}, a_{C_\infty} > 0$ and $b_{E_\infty}, b_{C_\infty} < 0$. At this point, we can apply the standard function scale, see Theorem 4, to the presentation of $E_\infty(z)$ and $C_\infty(z)$ in (24) and conclude that

$$[z^n]E_\infty(z) \sim C_{E_\infty}\rho^{-n}n^{-3/2} \quad \text{and} \quad [z^n]C_\infty(z) \sim C_{C_\infty}\rho^{-n}n^{-3/2} \tag{25}$$

where $C_{E_\infty} = \dfrac{b_{E_\infty}}{\Gamma(-\frac{1}{2})}$ and $C_{C_\infty} = \dfrac{b_{C_\infty}}{\Gamma(-\frac{1}{2})}$, respectively, with $\Gamma(-\frac{1}{2}) = 2\sqrt{\pi}$. In fact, reformulating (22) so to fit the Newton-Puiseux expansion forms (24) we find that

$$a_{E_\infty} = 2, \quad b_{E_\infty} = -\frac{1}{4}\sqrt{\frac{5}{47}\left(109 + 35\sqrt{545}\right)} \tag{26}$$

and

$$a_{C_\infty} = \frac{1}{2}, \quad b_{C_\infty} = \frac{2\sqrt{\frac{10\left(48069\sqrt{5}-10295\sqrt{109}\right)}{65\sqrt{109}-301\sqrt{5}}}}{3\sqrt{545}-77} \tag{27}$$

Numerical approximations of $C_{E_\infty} = \dfrac{b_{E_\infty}}{\Gamma(-\frac{1}{2})}$ and $C_{C_\infty} = \dfrac{b_{C_\infty}}{\Gamma(-\frac{1}{2})}$ yield the declared asymptotic behaviour of $(e_n)_n$ and $(c_n)_n$, see (18). ◀

Let us notice that as both generating functions $E_\infty(z)$ and $C_\infty(z)$ are algebraic, they are also *holonomic* (D-finite), i.e. satisfy differential equations with polynomial (in terms of $z$) coefficients. Using the powerful `gfun` library for `Maple` [34] one can automatically derive appropriate holonomic equations for $E_\infty(z)$ and $C_\infty(z)$, subsequently converting them into linear recurrences for sequences $(e_n)_n$ and $(c_n)_n$.

▶ **Example 10.** We restrict the presentation to the linear recurrence for the number of plain environments, omitting for brevity the, likely verbose, respective recurrence for plain closures. Using `gfun` we find that $e_n$ satisfies the recurrence of Figure 2. Despite its appearance, this recurrence is an efficient way of computing $e_n$. Indeed, holonomic specifications for $C_\infty(z)$ and $E_\infty(z)$ allow computing the coefficients $[z^n]C_\infty(z)$ and $[z^n]E_\infty(z)$ using a linear number of arithmetic operations, as opposed to a quadratic number of operations as following their direct combinatorial specification. Let us remark that the computations involved operate on large, having linear in $n$ space representation, integers. For instance, $e_{1000}$ has about 600 digits. In consequence, single arithmetic operations on such numbers cannot be performed in constant time.

▶ Remark. The analytic approach utilising generating functions exhibits an important benefit in the context of generating random instances of plain environments and closures. With analytic generating functions at hand and effective means of computing both $[z^n]E_\infty(z)$ and $[z^n]C_\infty(z)$, it is possible to design efficient samplers, constructing uniformly random (conditioned on the outcome size $n$) structures of both combinatorial classes. For instance, using holonomic specifications it becomes possible to construct exact-size samplers following the so-called recursive method of Nijenhuis and Wilf, see [31, 20]. Moreover, since corresponding generating functions are analytic, it is possible to design effective Boltzmann samplers [17], either in their approximate-size variant constructing structures within a structure size interval $[(1-\varepsilon)n, (1+\varepsilon)n]$ in time $O(|\omega|)$ where $\omega$ is the outcome structure, or their exact-size variants running in time $O(n^2)$. Remarkably, both sampler frameworks admit effective tuning procedures influencing the expected internal shape of constructed objects, e.g. frequencies of desired sub-patterns [5]. With the growing popularity of (semi-)automated software testing

$$
\begin{aligned}
&(125\,n^3 - 125\,n)\; e_n + \\
&(-475\,n^3 - 150\,n^2 + 325\,n)\; e_{n+1} + \\
&(-1625\,n^3 - 13650\,n^2 - 29125\,n - 17100)\; e_{n+2} + \\
&(5925\,n^3 + 65550\,n^2 + 204825\,n + 190800)\; e_{n+3} + \\
&(-10950\,n^3 - 149850\,n^2 - 609000\,n - 744300)\; e_{n+4} + \\
&(43599\,n^3 + 638460\,n^2 + 3028701\,n + 4633680)\; e_{n+5} + \\
&(-97781\,n^3 - 1680378\,n^2 - 9481237\,n - 17550960)\; e_{n+6} + \\
&(122749\,n^3 + 2388066\,n^2 + 15211685\,n + 31648968)\; e_{n+7} + \\
&(-184402\,n^3 - 3954630\,n^2 - 27717140\,n - 63149544)\; e_{n+8} + \\
&(280081\,n^3 + 6826380\,n^2 + 54868451\,n + 145130568)\; e_{n+9} + \\
&(-205649\,n^3 - 5654610\,n^2 - 51851989\,n - 158722620)\; e_{n+10} + \\
&(37439\,n^3 + 1339686\,n^2 + 16635271\,n + 70682784)\; e_{n+11} + \\
&(-68686\,n^3 - 3028038\,n^2 - 43616336\,n - 205972920)\; e_{n+12} + \\
&(222029\,n^3 + 9258780\,n^2 + 128417911\,n + 592399800)\; e_{n+13} + \\
&(-241115\,n^3 - 10519830\,n^2 - 152823475\,n - 739190880)\; e_{n+14} + \\
&(134151\,n^3 + 6201222\,n^2 + 95476551\,n + 489605640)\; e_{n+15} + \\
&(-42231\,n^3 - 2067834\,n^2 - 33729375\,n - 183277332)\; e_{n+16} + \\
&(7470\,n^3 + 386418\,n^2 + 6659316\,n + 38233296)\; e_{n+17} + \\
&(-678\,n^3 - 36972\,n^2 - 671670\,n - 4065240)\; e_{n+18} + \\
&(24\,n^3 + 1380\,n^2 + 26436\,n + 168720)\; e_{n+19} = 0.
\end{aligned}
$$

$$
\begin{aligned}
e_0 &= 1, \\
e_1 &= 1, \\
e_2 &= 4, \\
e_3 &= 17, \\
e_4 &= 77, \\
e_5 &= 364, \\
e_6 &= 1776, \\
e_7 &= 8881, \\
e_8 &= 45296, \\
e_9 &= 234806,
\end{aligned}
\qquad
\begin{aligned}
e_{10} &= 1233816, \\
e_{11} &= 6558106, \\
e_{12} &= 35202448, \\
e_{13} &= 190568779, \\
e_{14} &= 1039296373, \\
e_{15} &= 5704834700, \\
e_{16} &= 31494550253, \\
e_{17} &= 174759749005, \\
e_{18} &= 974155147162.
\end{aligned}
$$

**Figure 2** Linear recurrence defining $e_n$ with corresponding initial conditions.

techniques, see [12], combinatorial samplers for environments and closures exhibit potential applications in testing normalisation frameworks and abstract machines implementations, such as the Krivine machine. We briefly remark that randomly generated $\lambda$-terms already proved useful in finding optimisation bugs in compilers of functional programming languages, see [32]. Our prototype samplers for environments and closures, within above sampler frameworks, are available at Github[4].

## 6 Counting closed closures

In this section we address the problem of counting so-called *closed closures*[5]. A closure is said to be *closed* if it consists of an $m$-open term associated with an environment of length $m$ made itself out of closed closures. Note that such closures correspond to not yet fully evaluated $m$-open $\lambda$-terms. With such a description, the set $\mathcal{C}los_0$ of closed closures can be given using the following combinatorial specification:

$$\mathcal{C}los_0 ::= \mathcal{L}_0 \times \Box \;\mid\; \mathcal{L}_1 \times \langle \mathcal{C}los_0 \rangle \;\mid\; \mathcal{L}_2 \times \langle \mathcal{C}los_0, \mathcal{C}los_0 \rangle \;\mid\; \mathcal{L}_3 \times \langle \mathcal{C}los_0, \mathcal{C}los_0, \mathcal{C}los_0 \rangle \;\mid\; \cdots \quad (28)$$

[4] `https://github.com/PierreLescanne/CountingAndGeneratingClosuresAndEnvironments`

[5] We acknowledge that speaking of closed closures is a bit odd, however terms “closure” and “closed” form a consecrated terminology that we merely associate together.

▶ **Example 11.** The following table lists the first few closed closures.

| size | closures | total |
|---|---|---|
| $0, 1$ | | 0 |
| 2 | $\langle \lambda \underline{0}, \square \rangle$ | 1 |
| 3 | $\langle \lambda\lambda \underline{0}, \square \rangle \quad \langle \underline{0}, \langle \lambda \underline{0}, \square \rangle \rangle$ | 2 |
| 4 | $\langle \lambda\lambda\lambda \underline{0}, \square \rangle \quad \langle \lambda\lambda \underline{1}, \square \rangle \quad \langle \lambda(\underline{00}), \square \rangle$ $\langle \lambda \underline{0}, \langle \lambda \underline{0}, \square \rangle \rangle \quad \langle \underline{0}, \langle \lambda\lambda \underline{0}, \square \rangle \rangle \quad \langle \underline{0}, \langle \underline{0}, \langle \lambda \underline{0}, \square \rangle \rangle \rangle$ | 6 |

Establishing the asymptotic growth rate of the sequence $(c_{0,n})_n$ corresponding to closed closures of size $n$ poses a considerable challenge, much more involved than its plain counterpart. In the following theorem we show that there exists two constants $\underline{\rho}, \overline{\rho} < \rho_{L_\infty}$ such that $\lim_{n\to\infty} \frac{\underline{\rho}^{-n}}{c_{0,n}} = 0$ and $\lim_{n\to\infty} \frac{c_{0,n}}{\overline{\rho}^{-n}} = 0$. In other words, the asymptotic growth rate of $(c_{0,n})_n$ is bounded by two exponential functions of $n$.

▶ **Theorem 12.** *There exist $\overline{\rho} < \underline{\rho}$ satisfying $\overline{\rho} < \underline{\rho} < \rho_{L_\infty}$ and functions $\theta(n), \kappa(n)$ satisfying $\limsup_{n\to\infty} \theta(n)^{1/n} = \limsup_{n\to\infty} \kappa(n)^{1/n} = 1$ such that for sufficiently large $n$ we have $\underline{\rho}^{-n}\theta(n) < c_{0,n} < \overline{\rho}^{-n}\kappa(n)$.*

**Proof.** Let us start with the generating function $C_0(z)$ associated with closed closures $\mathcal{C}los_0$. Note that from the specification (28) $C_0(z)$ is implicitly defined as

$$C_0(z) = \sum_{m\geq 0} L_m(z) C_0(z)^m. \tag{29}$$

We can therefore identify a closed closure $\mathfrak{c}$ with a tuple $(t, c_1, \ldots, c_m)$ where $m \geq 0$, $t$ is an $m$-open $\lambda$-term and $c_1, \ldots, c_m$ are closed closures themselves. We proceed with defining two auxiliary lower and upper bound classes $\underline{C}_0(z)$ and $\overline{C}_0(z)$ such that $[z^n]\underline{C}_0(z) \leq [z^n]C_0(z) \leq \overline{C}_0(z)$ for all $n$. Next, we establish their asymptotic behaviour and, in doing so, provide exponential lower and upper bounds on the growth rate of closed closures.

We start with $\underline{C}_0(z) = \sum_{m\geq 0} L_0(z)\underline{C}_0(z)^m$. Note that $\underline{C}_0(z)$ is associated with closures in which each term is closed, independently of the corresponding environment length. Hence, as closed $\lambda$-terms are $m$-open for all $m \geq 0$, we have $[z^n]\underline{C}_0(z) \leq [z^n]C_0(z)$. Furthermore

$$\underline{C}_0(z) = \sum_{m\geq 0} L_0(z)\underline{C}_0(z)^m = L_0(z) \sum_{m\geq 0} \underline{C}_0(z)^m = \frac{L_0(z)}{1 - \underline{C}_0(z)}. \tag{30}$$

Solving the above equation for $\underline{C}_0(z)$ we find that $\underline{C}_0(z) = \frac{1}{2}\left(1 - \sqrt{1 - 4L_0(z)}\right)$. In such a form, it is clear that there are two potential sources of singularities, i.e. the singularity $\rho_{L_\infty}$ of $L_0(z)$, see Theorem 7, or the roots of the radicand $1 - 4L_0(z)$. Since $L_0(z)$ is increasing and continuous in the interval $(0, \rho_{L_\infty})$ we know that if $L_0(\rho_{L_\infty}) > \frac{1}{4}$ then there exists a $\underline{\rho} < \rho_{L_\infty}$ such that $L_0(\underline{\rho}) = \frac{1}{4}$. Unfortunately, we cannot simply check that $L_0(\rho) > \frac{1}{4}$ as there exists no known method of evaluating $L_0(z)$, defined by means of an infinite system of equations, at a given point. For that reason we propose the following approach.

Recall that a $\lambda$-term $M$ is said to be *$h$-shallow* if all its de Bruijn index values are (strictly) bounded by $h$, see [22]. Let $L_m^{(h)}(z)$ denote the generating function associated with $m$-open $h$-shallow $\lambda$-terms. Note that $L_0^{(h)}(z)$, i.e. the generating function corresponding to closed

$h$-shallow $\lambda$-terms, has a finite computable representation. Indeed, we have

$$\begin{aligned}
L_0^{(h)}(z) &= zL_1^{(h)}(z) + zL_0^{(h)}(z)L_0^{(h)}(z)\\
L_1^{(h)}(z) &= zL_2^{(h)}(z) + zL_1^{(h)}(z)L_1^{(h)}(z) + z\\
L_2^{(h)}(z) &= zL_3^{(h)}(z) + zL_2^{(h)}(z)L_2^{(h)}(z) + z + z^2\\
&\dots\\
L_{h-1}^{(h)}(z) &= zL_h^{(h)}(z) + zL_{h-1}^{(h)}(z)L_{h-1}^{(h)}(z) + z + z^2 + \dots + z^{h-1}\\
L_h^{(h)}(z) &= zL_h^{(h)}(z) + zL_h^{(h)}(z)L_h^{(h)}(z) + z + z^2 + \dots + z^h
\end{aligned} \tag{31}$$

Consider $m < h$. Each $m$-open $h$-shallow $\lambda$-term is either (a) in form of $\lambda M$ where $M$ is an $(m+1)$-open $h$-shallow $\lambda$-term due to the head abstraction, (b) in form of $MN$ where both $M$ and $N$ are $m$-open $h$-shallow $\lambda$-terms, or (c) a de Bruijn index in the set $\{\underline{0}, \underline{1}, \dots, \underline{m-1}\}$. When $m = h$, we have the same specification with the exception of the first summand $zL_h^{(h)}(z)$ where, as we cannot exceed $h$, terms under abstractions are $h$-open, instead of $(h+1)$-open.

Using such a form it is possible to evaluate $L_0^{(h)}(z)$ at each point $z \in (0, \rho_{(h)})$ where $\rho_{(h)} > \rho_{L_\infty}$ is the dominating singularity of $L_0^{(h)}(z)$ satisfying $\rho_{(h)} \xrightarrow[h\to\infty]{} \rho$, see [22]. Certainly, each closed $h$-shallow $\lambda$-term is in particular a closed $\lambda$-term. In consequence, $[z^n]L_0^{(h)}(z) \leq [z^n]L_0(z)$ for each $n$. Moreover, for all sufficiently large $n$ we have $[z^n]L_0^{(h)}(z) < [z^n]L_0(z)$. This coefficient-wise lower bound transfers onto the level of generating function values and we obtain $L_0^{(h)}(z) < L_0(z)$. Following the same argument, we also have $L_0^{(h)}(z) < L_0^{(h+1)}(z)$ for each $h \geq 1$. We can therefore use $L_0^{(h)}(z)$ to approximate $L_0(z)$ from below — the higher $h$ we choose, the better approximation we obtain. Using computer algebra software[6] it is possible to automatise the evaluation process of $L_0^{(h)}(\rho_{L_\infty})$ for increasing values of $h$ and find that for $h = 153$ we obtain

$$L_0^{(153)}(\rho_{L_\infty}) \doteq 0.25000324068941554\,. \tag{32}$$

Hence indeed, the asserted existence of $\underline{\rho} < \rho_{L_\infty}$ such that $L_0(\underline{\rho}) = \frac{1}{4}$ follows (interestingly, taking $h = 152$ does not suffice as $L_0^{152}(\rho_{L_\infty}) < \frac{1}{4}$). We fall hence in the super-critical composition schema[7] and note that $\underline{C}_0(z)$ admits a Newton-Puiseux expansion near $\underline{\rho}$ as follows:

$$\underline{C}_0(z) = \underline{a_0} - \underline{b_0}\sqrt{1 - \frac{z}{\underline{\rho}}} + O\left(\left|1 - \frac{z}{\underline{\rho}}\right|\right) \tag{33}$$

for some constants $\underline{a_0} > 0$ and $\underline{b_0} < 0$. Hence, $[z^n]C_0(z)$ grows asymptotically faster than $\underline{\rho}^{-n}\theta(n)$ where $\theta(n) = \dfrac{\underline{b_0}}{\Gamma(-\frac{1}{2})}n^{-3/2}$.

For the upper bound we consider $\overline{C}_0(z) = \sum_{m\geq 0} L_\infty(z)\overline{C}_0(z)^m$, i.e. the generating function associated with closures in which all terms are plain (either closed or open), independently of the constraint imposed by the corresponding environment length. Following the same arguments as before, we note that $[z^n]\overline{C}_0(z) > [z^n]C_0(z)$. Now

$$\overline{C}_0(z) = \sum_{m\geq 0} L_\infty(z)\overline{C}_0(z)^m = L_\infty(z)\sum_{m\geq 0}\overline{C}_0(z)^m = \frac{L_\infty(z)}{1 - \overline{C}_0(z)}. \tag{34}$$

[6] `https://github.com/PierreLescanne/CountingAndGeneratingClosuresAndEnvironments`

[7] *Supercriticality* ensures that meromorphic asymptotics applies and entails strong statistical regularities (see [19] Section V.2 and Section IX.6).

Solving the equation for $\overline{C}_0(z)$ we find that $\overline{C}_0(z) = \frac{1}{2}\left(1 - \sqrt{1 - 4L_\infty(z)}\right)$. Note that in this case, we can easily handle the radicand expression $1 - 4L_\infty(z)$ and find out that, as in the lower bound case, we are in the super-critical composition schema. Specifically, $\overline{\rho} = \frac{1}{10}\left(25 - \sqrt{545}\right) \doteq 0.165476$, cf. (20), is the dominating singularity of $\overline{C}_0(z)$. In consequence, $\overline{C}_0(z)$ admits the following Newton-Puiseux expansion near $\overline{\rho}$:

$$\overline{C}_0(z) = \overline{a_0} - \overline{b_0}\sqrt{1 - \frac{z}{\overline{\rho}}} + O\left(\left|1 - \frac{z}{\overline{\rho}}\right|\right) \tag{35}$$

for some constants $\overline{a_0} > 0$ and $\overline{b_0} < 0$. In conclusion, $[z^n]C_0(z)$ grows asymptotically slower than $(\overline{\rho})^{-n}\theta(n)$ where $\theta(n) = \dfrac{\overline{b_0}}{\Gamma(-\frac{1}{2})}n^{-3/2}$, finishing the proof. ◀

With an implicit expression defining $C_0(z)$, see (29), efficient random generation of closed closures poses a difficult task. Though we have no efficient Boltzmann samplers, it is possible to follow the recursive method and an obtain exact-size samplers for a moderate range of target sizes. We offer a prototype sampler of this kind, available at Github[8].

## 7 Conclusions

We view our contribution as a small step towards the quantitative, average-case analysis of evaluation complexity in $\lambda$-calculus. Using standard tools from analytic combinatorics, we investigated some combinatorial aspects of environments and closures — fundamental structures present in various formalisms dealing with normalisation in $\lambda$-calculus, especially in its variants with explicit substitutions [28]. Though plain environments and closures are relatively easy to count and generate, their closed counterparts pose a considerable combinatorial challenge. The implicit and infinite specification of closed closures based on closed $\lambda$-terms complicates significantly the quantitative analysis, namely estimating the exponential factor in the asymptotic growth rate, or effectively generating random closed closures. In particular, getting more parameters of the asymptotic growth will require more sophisticated methods, like, for instance, the recent infinite system approximation techniques of Bodini, Gittenberger and Gołębiewski [9].

## References


1 Beniamino Accattoli and Ugo Dal Lago. (leftmost-outermost) beta reduction is invariant, indeed. *Logical Methods in Computer Science*, 12(1), 2016. doi:10.2168/LMCS-12(1:4)2016.

2 Martin Avanzini, Ugo Dal Lago, and Georg Moser. Analysing the complexity of functional programs: higher-order meets first-order. In Kathleen Fisher and John H. Reppy, editors, *Proceedings of the 20th ACM SIGPLAN International Conference on Functional Programming, ICFP 2015, Vancouver, BC, Canada.*, pages 152–164. ACM, 2015. doi:10.1145/2784731.2784753.

3 Martin Avanzini and Georg Moser. Closing the gap between runtime complexity and polytime computability. In Christopher Lynch, editor, *Proceedings of the 21st International Conference on Rewriting Techniques and Applications, RTA 2010, July 11-13, 2010, Edinburgh, Scottland, UK*, volume 6 of *LIPIcs*, pages 33–48. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010. doi:10.4230/LIPIcs.RTA.2010.33.


[8] https://github.com/PierreLescanne/CountingAndGeneratingClosuresAndEnvironments

4 Maciej Bendkowski. Normal-order reduction grammars. *Journal of Functional Programming*, 27, 2017. doi:10.1017/S0956796816000332.

5 Maciej Bendkowski, Olivier Bodini, and Sergey Dovgal. *Polynomial tuning of multiparametric combinatorial samplers*, pages 92–106. SIAM, 2018. doi:10.1137/1.9781611975062.9.

6 Maciej Bendkowski, Katarzyna Grygiel, Pierre Lescanne, and Marek Zaionc. Combinatorics of $\lambda$-terms: a natural approach. *Journal of Logic and Computation*, 27(8):2611–2630, 2017. doi:10.1093/logcom/exx018.

7 Maciej Bendkowski, Katarzyna Grygiel, and Marek Zaionc. On the likelihood of normalization in combinatory logic. *Journal of Logic and Computation*, 2017. doi:10.1093/logcom/exx005.

8 Olivier Bodini, Danièle Gardy, and Bernhard Gittenberger. Lambda-terms of bounded unary height. In Philippe Flajolet and Daniel Panario, editors, *Proceedings of the Eighth Workshop on Analytic Algorithmics and Combinatorics, ANALCO 2011, San Francisco, California, USA, January 22, 2011*, pages 23–32. SIAM, 2011. doi:10.1137/1.9781611973013.3.

9 Olivier Bodini, Bernhard Gittenberger, and Zbigniew Gołębiewski. Enumerating lambda terms by weighted length of their de bruijn representation. *CoRR*, abs/1707.02101, 2017. URL: https://arxiv.org/abs/1707.02101.

10 Christine Choppy, Stéphane Kaplan, and Michèle Soria. Algorithmic complexity of term rewriting systems. In Pierre Lescanne, editor, *Rewriting Techniques and Applications, 2nd International Conference, RTA-87, Bordeaux, France, May 25-27, 1987, Proceedings*, volume 256 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 1987. doi:10.1007/3-540-17220-3_22.

11 Christine Choppy, Stéphane Kaplan, and Michèle Soria. Complexity analysis of term-rewriting systems. *Theor. Comput. Sci.*, 67(2&3):261–282, 1989. doi:10.1016/0304-3975(89)90005-4.

12 Koen Claessen and John Hughes. Quickcheck: A lightweight tool for random testing of haskell programs. In *Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming*, pages 268–279. ACM, 2000.

13 Pierre-Louis Curien. *Categorical Combinators, Sequential Algorithms, and Functional Programming (2nd Ed.)*. Birkhauser Boston Inc., Cambridge, MA, USA, 1994.

14 Pierre-Louis Curien, Thérèse Hardin, and Jean-Jacques Lévy. Confluence properties of weak and strong calculi of explicit substitutions. *Journal of the ACM*, 43(2):362–397, March 1996. doi:10.1145/226643.226675.

15 René David, Katarzyna Grygiel, Jakub Kozik, Christophe Raffalli, Guillaume Theyssier, and Marek Zaionc. Asymptotically almost all $\lambda$-terms are strongly normalizing. *Logical Methods in Computer Science*, 9:1–30, 2013.

16 Nicolaas G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Indagationes Mathematicae (Proceedings)*, 75(5):381–392, 1972.

17 Philippe Duchon, Philippe Flajolet, Guy Louchard, and Gilles Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability and Computing*, 13(4-5):577–625, 2004.

18 Philippe Flajolet and Andrew M. Odlyzko. Singularity analysis of generating functions. *SIAM Journal on Discrete Mathematics*, 3(2):216–240, 1990.

19 Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 1 edition, 2009.

20 Philippe Flajolet, Paul Zimmermann, and Bernard Van Cutsem. A calculus for the random generation of labelled combinatorial structures. *Theoretical Computer Science*, 132(1):1–35, 1994.

21 Étienne Ghys. *A singular mathematical promenade*. Ecole Normale Supérieure, 2017. URL: http://perso.ens-lyon.fr/ghys/promenade/.

22 Bernhard Gittenberger and Zbigniew Gołębiewski. On the number of lambda terms with prescribed size of their de Bruijn representation. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS*, pages 40:1–40:13, 2016.

23 Katarzyna Grygiel and Pierre Lescanne. Counting and generating terms in the binary lambda calculus. *Journal of Functional Programming*, 25, 2015. doi:10.1017/S0956796815000271.

24 Donald E. Knuth. Mathematical Analysis of Algorithms, 2000. First chapter of [25].

25 Donald E. Knuth. *Selected Papers on Analysis of Algorithms*, volume 102 of *CSLI Lecture Notes*. Stanford, California: Center for the Study of Language and Information, 2000.

26 Ugo Dal Lago and Simone Martini. On constructor rewrite systems and the lambda calculus. *Logical Methods in Computer Science*, 8(3), 2012. doi:10.2168/LMCS-8(3:12)2012.

27 Peter J. Landin. The mechanical evaluation of expressions. *The Computer Journal*, 6(4):308–320, 1964. doi:10.1093/comjnl/6.4.308.

28 Pierre Lescanne. From $\lambda\sigma$ to $\lambda\upsilon$: A journey through calculi of explicit substitutions. In *Proceedings of the 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 60–69. ACM, 1994.

29 Michel Mauny and Ascánder Suárez. Implementing functional languages in the categorical abstract machine. In *LISP and Functional Programming*, pages 266–278, 1986.

30 John C. Mitchell. *Concepts in Programming Language (1st Ed.)*. Cambridge University Press, New York, NY, USA, 2002.

31 Albert Nijenhuis and Herbert S. Wilf. *Combinatorial Algorithms*. Academic Press, 2 edition, 1978.

32 Michał H. Pałka. *Random Structured Test Data Generation for Black-Box Testing*. PhD thesis, Chalmers University of Technology, 2012.

33 Gordon David Plotkin. Call-by-name, call-by-value and the $\lambda$-calculus. *Theoretical Computer Science*, 1(2):125–159, 1975. doi:10.1016/0304-3975(75)90017-1.

34 Bruno Salvy and Paul Zimmermann. Gfun: a Maple package for the manipulation of generating and holonomic functions in one variable. *ACM Transactions on Mathematical Software*, 20(2):163–177, 1994.

35 Robert Sedgewick and Philippe Flajolet. *An Introduction to the Analysis of Algorithms (2nd Edition)*. Createspace Independent Pub, 2014.

36 Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51(3):385–463, 2004. doi:10.1145/990308.990310.

37 Herbert S. Wilf. *Generatingfunctionology*. A. K. Peters, Ltd., 2006.